```
  AAAAAA      CCCCCCCC  TTTTTTTTTT  CCCCCCCC  HH      HH  RRRRRRR
  AAAAAA      CCCCCCCC  TTTTTTTTTT  CCCCCCCC  HH      HH  RRRRRRR
AA      AA  CC            TT      CC          HH      HH  RR      RR
AA      AA  CC            TT      CC          HH      HH  RR      RR
AA      AA  CC            TT      CC          HH      HH  RR      RR
AA      AA  CC            TT      CC          HHHHHHHHHH  RRRRRRR
AAAAAAAAAA  CC            TT      CC          HHHHHHHHHH  RRRRRRR
AAAAAAAAAA  CC            TT      CC          HH      HH  RR RR
AA      AA  CC            TT      CC          HH      HH  RR  RR
AA      AA  CC            TT      CC          HH      HH  RR   RR    ....
AA      AA    CCCCCCCC    TT        CCCCCCCC  HH      HH  RR    RR   ....
AA      AA    CCCCCCCC    TT        CCCCCCCC  HH      HH  RR    RR   ....
```

```
LL          IIIIII    SSSSSSSS
LL          IIIIII    SSSSSSSS
LL            II    SS
LL            II    SS
LL            II    SS
LL            II      SSSSSS
LL            II      SSSSSS
LL            II          SS
LL            II          SS
LL            II          SS
LL            II          SS
LLLLLLLLLL  IIIIII    SSSSSSSS
LLLLLLLLLL  IIIIII    SSSSSSSS
```

MAC$ACTCHR
Table of contents
CHARACTER STRING ROUTINES
I 12
16-SEP-1984 01:58:23 VAX/VMS Macro V04-00
Page 0

MAC$ACTCHR
V04-000

CHARACTER STRING ROUTINES

J 12

16-SEP-1984 01:58:23   VAX/VMS Macro V04-00   Page 1
5-SEP-1984 01:46:46   [MACRO.SRC]ACTCHR.MAR;1   (1)

```
0000    1              .TITLE  MAC$ACTCHR      CHARACTER STRING ROUTINES
0000    2              .IDENT  'V04-000'
0000    3
0000    4      ;
0000    5      ;************************************************************************
0000    6      ;*                                                                      *
0000    7      ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                             *
0000    8      ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.              *
0000    9      ;*  ALL RIGHTS RESERVED.                                                *
0000   10      ;*                                                                      *
0000   11      ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED *
0000   12      ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE *
0000   13      ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER *
0000   14      ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY *
0000   15      ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY *
0000   16      ;*  TRANSFERRED.                                                         *
0000   17      ;*                                                                      *
0000   18      ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE *
0000   19      ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT *
0000   20      ;*  CORPORATION.                                                         *
0000   21      ;*                                                                      *
0000   22      ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS *
0000   23      ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.              *
0000   24      ;*                                                                      *
0000   25      ;*                                                                      *
0000   26      ;************************************************************************
0000   27      ;
0000   28
0000   29      ;++
0000   30      ; FACILITY:     VAX MACRO ASSEMBLER OBJECT LIBRARY
0000   31      ;
0000   32      ; ABSTRACT:
0000   33      ;
0000   34      ; The VAX-11 MACRO assembler translates MACRO-32 source code into object
0000   35      ; modules for input to the VAX-11 LINKER.
0000   36      ;
0000   37      ; ENVIRONMENT:  USER MODE
0000   38      ;
0000   39      ; AUTHOR: Benn Schreiber, CREATION DATE: 21-AUG-78
0000   40      ;
0000   41      ; MODIFIED BY:
0000   42      ;
0000   43      ;       V03-001 ROP0026         Robert Posniak  20-JUL-1984
0000   44      ;               Make sure check for continuation line on
0000   45      ;               .ASCID.  SPR # 11-60136
0000   46      ;
0000   47      ;       V02.02  MTR0001         Mike Rhodes     02-Feb-1982
0000   48      ;               Set FLG$V_DLIMSTR to pass ALL characters in a
0000   49      ;               delimited .ASCIx string.  Fix for QAR #890 and
0000   50      ;               SPR #11-42904.
0000   51      ;
0000   52      ;       V02.01  PCG0001         Peter George    10-Apr-1981
0000   53      ;               Clear registers and modes after NTYPE.
0000   54      ;
0000   55      ;       V01.12  RN0023          R. Newland      2-Nov-1979
0000   56      ;               New message codes to get error message from system
0000   57      ;               message file.
```

MACSACTCHR
V04-000

CHARACTER STRING ROUTINES

K 12

16-SEP-1984 01:58:23 VAX/VMS Macro V04-00
5-SEP-1984 01:46:46 [MACRO.SRC]ACTCHR.MAR;1

Page 2
(1)

```
0000    58 ;
0000    59 ;        V01.11  RN0016          R. Newland      19-Oct-1979
0000    60 ;                Don't output error messages when .NTYPE operand
0000    61 ;                argument is the PC.  SPR 11-26392
0000    62 ;
0000    63 ;        V01.10  RN0005          R. Newland      26-Aug-1979
0000    64 ;                Reorder macro definitions and remove .ALIGN LONG
0000    65 ;                and .DEBUG statements
0000    66 ;
0000    67 ;--
```

MAC$ACTCHR   CHARACTER STRING ROUTINES   L 12   16-SEP-1984 01:58:23   VAX/VMS Macro V04-00   Page 3
V04-000   DECLARATIONS   5-SEP-1984 01:46:46   [MACRO.SRC]ACTCHR.MAR;1   (2)

```
                  0000    69              .SBTTL  DECLARATIONS
                  0000    70 ;
                  0000    71 ; INCLUDE FILES:
                  0000    72 ;
                  0000    73
                  0000    74 ;
                  0000    75 ; MACROS:
                  0000    76 ;
                  0000    77
                  0000    78              $MAC_INTCODDEF                  ;DEFINE INT. FILE CODES
                  0000    79              $MAC_GENVALDEF                  ;DEFINE GENERAL VALUES
                  0000    80              $MAC_CTLFLGDEF                  ;DEFINE CONTROL FLAGS
                  0000    81              $MAC_SYMBLKDEF                  ;DEFINE SYMBOL BLOCK OFFSETS
                  0000    82              $MAC_INPBLKDEF                  ;DEFINE INPUT BLOCK OFFSETS
                  003C    83              $MAC_GRAMMARDEF                 ;DEFINE TERMINAL GRAMMER SYMBOLS
                  003C    84              $DSCDEF                         ;DEFINE ARG DESCRIPTORS
                  003C    85              $MACMSGDEF                      ; Define message codes
                  003C    86
                  003C    87 ;
                  003C    88 ; EQUATED SYMBOLS
                  003C    89 ;
                  003C    90
         00000000 003C    91              ASC$_ASCIC       =       0      ;ASCIC CODE
         00000001 003C    92              ASC$_ASCID       =       1      ;ASCID
         00000002 003C    93              ASC$_ASCII       =       2      ;ASCII
         00000003 003C    94              ASC$_ASCIZ       =       3      ;ASCIZ
                  003C    95
         00000000         96              .PSECT  MAC$RO_CODE_P1,NOWRT,GBL,LONG
```

MAC$ACTCHR
V04-000

M 12
CHARACTER STRING ROUTINES   16-SEP-1984 01:58:23   VAX/VMS Macro V04-00   Page   4
CHARACTER DATA GENERATING DIRECTIVES   5-SEP-1984 01:46:46   [MACRO.SRC]ACTCHR.MAR;1   (3)

```
              0000      98                    .SBTTL   CHARACTER DATA GENERATING DIRECTIVES
              0000      99
              0000     100 ;++
              0000     101 ; FUNCTIONAL DESCRIPTION:
              0000     102 ;
              0000     103 ;        ASCIC IS CALLED WHEN A .ASCIC DIRECTIVE IS FOUND.  THE ASCII
              0000     104 ;        STRING IS EMITTED WITH A PRECEDING COUNT BYTE
              0000     105 ;
              0000     106 ;--
              0000     107
              0000     108 ASCIC::                                       ;CHAR_HEAD = KASCIC
              0000     109           $INTOUT_LW INT$_STKL,#0             ;STACK A 0
              0008     110           $INTOUT_X INT$_STSB                 ;STORE SIGNED BYTE
              000E     111           $INC_PC                             ;LEAVE ROOM FOR COUNT BYTE
        44 10 0012     112           BSBB      ASC_COM                   ;JOIN COMMON CODE
        00    0014     113           .BYTE     ASC$_ASCIC                ;THIS IS INDEX FOR ASCIC
              0015     114
              0015     115 ;++
              0015     116 ; FUNCTIONAL DESCRIPTION:
              0015     117 ;
              0015     118 ;        ASCID IS CALLED WHEN A .ASCID DIRECTIVE IS FOUND.  THE
              0015     119 ;        ASCII STRING IS EMITTED WITH A PRECEDING STRING DESCRIPTOR
              0015     120 ;        CONSISTING OF '.LONG STRING_LENGTH,.+1'.
              0015     121 ;
              0015     122 ;--
              0015     123
              0015     124 ASCID::
  00 6B  14 E3 0015     125           BBCS      #FLG$V_CHKLPND,(R11),.+1 ; Make sure we check for a continuation
              0019     126           $INTOUT_LW INT$_STKL,#<<1@24>+<DSC$K_DTYPE_T@16>> ;START STRING DESCRIPTOR
              0025     127           $INTOUT_X  INT$_STOL                ;AND STORE IT
              002B     128           $INC_PC #4                          ;COUNT 4 BYTES
              0030     129           $INTOUT_X  INT$_STKPC               ;STACK CURRENT PC
              0036     130           $INTOUT_LW INT$_STKL,#4             ;4 BYTES PLUS 1
              003E     131           $INTOUT_X  INT$_ADD                 ;POINT TO THE START OF THE STRING
              0044     132           $INTOUT_X  INT$_SPID                ;CALL IT PID?
              004A     133           $INC_PC #4                          ;COUNT 4 MORE BYTES
        07 10 004F     134           BSBB      ASC_COM                   ;JOIN COMMON CODE
        01    0051     135           .BYTE     ASC$_ASCID                ;INDEX FOR ASCID
              0052     136
              0052     137 ;++
              0052     138 ; FUNCTIONAL DESCRIPTION:
              0052     139 ;
              0052     140 ;        ASCII IS CALLED TO PROCESS A .ASCII DIRECTIVE. THE ASCII
              0052     141 ;        STRING IS SCANNED AND CODE IS  EMITTED TO PASS 2 TO EMIT
              0052     142 ;        THE STRING.
              0052     143 ;
              0052     144 ;--
              0052     145
              0052     146 ASCII::                                       ;CHAR_HEAD = KASCII
        04 10 0052     147           BSBB      ASC_COM                   ;JOIN COMMON CODE
        02    0054     148           .BYTE     ASC$_ASCII                ;INDEX FOR ASCII
              0055     149
              0055     150 ;++
              0055     151 ; FUNCTIONAL DESCRIPTION:
              0055     152 ;
              0055     153 ;        ASCIZ IS CALLED WHEN A .ASCIZ DIRECTIVE IS SCANNED.  THE
              0055     154 ;        ASCII STRING IS EMITTED WITH A ZERO BYTE AT THE END.
```

MAC$ACTCHR
V04-000

N 12

CHARACTER STRING ROUTINES   16-SEP-1984 01:58:23 VAX/VMS Macro V04-00   Page 5
CHARACTER DATA GENERATING DIRECTIVES 5-SEP-1984 01:46:46 [MACRO.SRC]ACTCHR.MAR;1   (3)

```
                         0055    155 ;
                         0055    156 ;--
                         0055    157
                         0055    158 ASCIZ::                                           ;CHAR_HEAD = KASCIZ
              01    10   0055    159         BSBB    ASC_COM                           ;JOIN COMMON CODE
                    03   0057    160         .BYTE   ASC$_ASCIZ                        ;INDEX FOR ASCIZ
                         0058    161 ;
                         0058    162 ; COMMON CODE FOR ASCIC/ASCID/ASCII/ASCIZ
                         0058    163 ;
                         0058    164 ASC_COM:
   0000'CF   9E    9A    0058    165         MOVZBL  @(SP)+,W^MAC$GL_DIRFLG            ;SET ASCIX INDEX
         0D   5A    91   005D    166         CMPB    R10,#CR                           ;WAS THERE NO DELIMITER?
               11   13   0060    167         BEQL    10$                               ;IF EQL YES--SYNTAX ERROR
   00 6B    2F    E3     0062    168         BBCS    #FLG$V_DLIMSTR,(R11),.+1          ;PASS ALL CHARACTERS IN STR.
      0000'CF   D4        0066    169         CLRL    W^MAC$GL_ASCCNT                   ;CLEAR  CHARACTER COUNT
      0000'CF   D7        006A    170         DECL    W^MAC$GL_LINEPT                   ;BACKUP TO REREAD DELIMITER
         FF8F'   30        006E    171         BSBW    MAC$GETCHR                        ;REREAD DELIMITER
               41   11    0071    172         BRB     GET_CHARS                         ;START SCANNING ASCII STRING
                         0073    173 10$:     $MAC_ERR DIRSYNX                          ; Report syntax error
         FF85'   31        0078    174         BRW     MAC$ERRORPT                       ;AND RETURN
```

MAC$ACTCHR
V04-000

B 13

CHARACTER STRING ROUTINES
CHARACTER DATA GENERATING DIRECTIVES

16-SEP-1984 01:58:23   VAX/VMS Macro V04-00   Page   6
5-SEP-1984 01:46:46   [MACRO.SRC]ACTCHR.MAR;1   (4)

```
                          007B    176  ;++
                          007B    177  ; FUNCTIONAL DESCRIPTION:
                          007B    178  ;
                          007B    179  ;        CHRNUL IS CALLED WHEN A NULL CHARACTER '<>' IS SEEN WHILE
                          007B    180  ;        SCANNING AN ASCIX DIRECTIVE.  A NULL BYTE IS EMITTED AND
                          007B    181  ;        SCANNING CONTINUES.
                          007B    182  ;
                          007B    183  ;--
                          007B    184
                          007B    185  CHRNUL::                              ;CHAR_ARGS = DANGOPN DANGCLS
                          007B    186                                        ;CHAR_ARGS = CHAR_ARGS DANGOPN DANGCLS
            50    D4      007B    187         CLRL     R0                    ;SET TO STORE NULL IMMED. BYTE
          FF80'   30      007D    188         BSBW     MAC$INTOUT_BY         ;STORE IMMEDIATE BYTE
                          0080    189         $INC_PC                        ;COUNT THE CHARACTER
         0000'CF  D6      0084    190         INCL     W^MAC$GL_ASCCNT       ;...
            2A    11      0088    191         BRB      GET_CHARS             ;CONTINUE SCANNING STRING
                          008A    192
                          008A    193  ;++
                          008A    194  ; FUNCTIONAL DESCRIPTION:
                          008A    195  ;
                          008A    196  ;        CHRARG IS CALLED WHEN AN EXPRESSION IN ANGLE BRACKETS IS
                          008A    197  ;        ENCOUNTERED WHILE SCANNING AN ASCIX STRING.  WHEN THE
                          008A    198  ;        ROUTINE 'GET_CHARS' FINDS AN ANGLE BRACKET, IT RETURNS TO
                          008A    199  ;        THE PARSER.  AFTER THE EXPRESSION IN ANGLE BRACKETS IS
                          008A    200  ;        SCANNED, THIS ROUTINE IS CALLED TO STORE THE EXPRESSION.
                          008A    201  ;        SCANNING OF THE STRING THEN CONTINUES UNTIL END-OF-LINE.
                          008A    202  ;
                          008A    203  ;--
                          008A    204
                          008A    205  CHRARG::                              ;CHAR_ARGS = DANGOPN EXPR DANGCLS
                          008A    206                                        ;CHAR_ARGS = CHAR_ARGS DANGOPN EXPR DANGCLS
         0000'CF  D5      008A    207         TSTL     W^MAC$GL_ABSFLAG      ;ABSOLUTE EXPRESSION?
            04    13      008E    208         BEQL     10$                   ;IF EQL YES
      12 6B   07  E4      0090    209         BBSC     #FLG$V_EXPOPT,(R11),20$ ;NO--DO NOT ALLOW EXPRESSION OPTIMIZATION
      0E 6B   07  E1      0094    210  10$:   BBC      #FLG$V_EXPOPT,(R11),20$ ;BRANCH IF CANNOT OPTIMIZE
          FF65'   30      0098    211         BSBW     MAC$OPTIMIZEXPR       ;YES--DO IT
   50 FFFC'CF47   D0      009B    212         MOVL     W^MAC$AL_VALSTACK-4[R7],R0 ;SET TO STORE IMMED. BYTE
          FF5C'   30      00A1    213         BSBW     MAC$INTOUT_BY         ;STORE IMMEDIATE BYTE
            06    11      00A4    214         BRB      30$                   ;CONTINUE
                          00A6    215  20$:   $INTOUT_X INT$_STOB            ;STORE BYTE
                          00AC    216  30$:   $INC_PC                        ;COUNT THE BYTE
         0000'CF  D6      00B0    217         INCL     W^MAC$GL_ASCCNT       ;...
                          00B4    218  ;**;  BRB      GET_CHARS             ;CONTINUE SCANNING STRING
```

```
                          00B4   220  ;++
                          00B4   221  ; FUNCTIONAL DESCRIPTION:
                          00B4   222  ;
                          00B4   223  ;       GET CHARS IS A LOCAL ROUTINE TO SCAN ASCIX STRINGS.  ALL
                          00B4   224  ;       CHARACTERS ARE PASSED.  THE STRING IS SCANNED UNTIL AN
                          00B4   225  ;       END-OF-LINE.  IF A BRACKETED EXPRESSION IS FOUND, GET_CHARS
                          00B4   226  ;       RETURNS TO THE PARSER TO GATHER THE EXPRESSION WITHIN THE
                          00B4   227  ;       ANGLE BRACKETS, AND THEN CALL CHRARG TO STORE THE EXPRESSION
                          00B4   228  ;       VALUE AND CONTINUE THE SCAN.
                          00B4   229  ;
                          00B4   230  ;--
                          00B4   231
                          00B4   232  GET_CHARS:
      00 6B   06   E3     00B4   233          BBCS    #FLG$V_EVALEXPR,(R11),.+1 ;ALLOW EXPRESSION EVALUATION
             FF45'  30    00B8   234  20$:    BSBW    MAC$SKIPSP             ;IGNORE BLANKS
         0D   5A   91     00BB   235          CMPB    R10,#CR               ;CARRIAGE RETURN?
         53   13          00BE   236          BEQL    70$                   ;IF EQL YES
      3C   5A   91        00C0   237          CMPB    R10,#^A/</            ;NO--BRACKETED EXPRESSION?
         4E   13          00C3   238          BEQL    70$                   ;IF EQL YES
      7E   5A   90        00C5   239          MOVB    R10,-(SP)             ;NO--SAVE DELIMITER
      00 6B   2F   E3     00C8   240          BBCS    #FLG$V_DLIMSTR,(R11),.+1 ;PASS ALL CHARS IN DLIM STR
      00 6B   00   E3     00CC   241          BBCS    #FLG$V_ALLCHR,(R11),30$ ;PASS ALL CHARACTERS
             FF2D'  30    00D0   242  30$:    BSBW    MAC$GETCHR            ;GET NEXT CHARACTER
      6E   5A   91        00D3   243          CMPB    R10,(SP)              ;MATCHING DELIMITER?
         0D   12          00D6   244          BNEQ    40$                   ;IF NEQ NO
      00 6B   00   E4     00D8   245          BBSC    #FLG$V_ALLCHR,(R11),.+1 ;YES--CLEAR ALL CHARS FLAG
      00 6B   2F   E4     00DC   246          BBSC    #FLG$V_DLIMSTR,(R11),.+1 ; NO MORE HYPHENS AND SEMIS
             FF1D'  30    00E0   247          BSBW    MAC$GETCHR            ;GET NEXT CHARACTER
         23   11          00E3   248          BRB     60$                   ;AND EXIT
      0D   5A   91        00E5   249  40$:    CMPB    R10,#CR               ;END OF LINE?
         0A   12          00E8   250          BNEQ    50$                   ;IF NEQ NO
                          00EA   251          $MAC_ERR UNTERMARG            ; Yes--get message code
             FF0E'  30    00EF   252          BSBW    MAC$ERRORLN          ;REPORT ERROR TO PASS 2
         14   11          00F2   253          BRB     60$                   ;EXIT
      50   5A   D0        00F4   254  50$:    MOVL    R10,R0                ;SET TO STORE IMMED. BYTE
             FF06'  30    00F7   255          BSBW    MAC$INTOUT_BY        ;STORE IMMEDIATE BYTE
                          00FA   256          $INC_PC                       ;UP THE PC
      0000'CF   D6        00FE   257          INCL    W^MAC$GL_ASCCNT      ;COUNT THE CHARACTER
      00 6B   2F   E3     0102   258          BBCS    #FLG$V_DLIMSTR,(R11),.+1 ;ALLOW HYPHENS AND SEMIS
         C8   11          0106   259          BRB     30$                   ;CONTINUE SCANNING
                          0108   260  ;
                          0108   261  ; HERE IF CLOSING DELIMITER SEEN OR IF WE FOUND CR BEFORE CLOSING
                          0108   262  ; DELIMITER
                          0108   263  ;
         5E   D6          0108   264  60$:    INCL    SP                    ;KEEP THE STACK STRAIGHT
      00 6B   00   E5     010A   265          BBCC    #FLG$V_ALLCHR,(R11),65$ ;DO NOT PASS ALL CHARACTERS
         0D   5A   91     010E   266  65$:    CMPB    R10,#CR               ;END OF LINE?
         A5   12          0111   267          BNEQ    20$                   ;IF NEQ NO--KEEP SCANNING
      00 6B   2F   E5     0113   268  70$:    BBCC    #FLG$V_DLIMSTR,(R11),.+1 ;NO MORE HYPHENS AND SEMIS
      00 6B   07   E3     0117   269          BBCS    #FLG$V_EXPOPT,(R11),.+1 ;ALLOW EXPRESSION OPTIMIZATION
      00 6B   00   E5     011B   270          BBCC    #FLG$V_ALLCHR,(R11),.+1 ;DO NOT PASS SEMI COLONS ANY MORE
      0000'CF   59   D0   011F   271          MOVL    R9,W^MAC$GL_EXPPTR   ;RESET EXPRESSION POINTERS
      0000'CF   59   D0   0124   272          MOVL    R9,W^MAC$GL_EXPEND   ;...
         0000'CF   D4     0129   273          CLRL    W^MAC$GL_ABSFLAG     ;ASSUME ABSOLUTE EXPRESSION
         0000'CF   D4     012D   274          CLRL    W^MAC$GL_PRMSEG      ;NO SEGMENT FOR EXPRESSION
         05              0131   275          RSB
```

```
                                    0132   277  ;++
                                    0132   278  ; FUNCTIONAL DESCRIPTION:
                                    0132   279  ;
                                    0132   280  ;        CHARHD IS CALLED WHEN A .ASCIX DIRECTIVE HAS BEEN COMPLETELY
                                    0132   281  ;        SCANNED.  ACTION TAKEN:
                                    0132   282  ;
                                    0132   283  ;        .ASCIC - EMIT THE CHARACTER COUNT BYTE
                                    0132   284  ;        .ASCID - EMIT THE CHARACTER STRING DESCRIPTOR
                                    0132   285  ;        .ASCII - NONE
                                    0132   286  ;        .ASCIZ - EMIT A ZERO BYTE AT THE END OF THE STRING
                                    0132   287  ;
                                    0132   288  ;--
                                    0132   289
                                    0132   290  CHARHD::                                      ;CHAR_STAT = CHAR_HEAD
                                    0132   291                                                ;CHAR_STAT = CHAR_HEAD CHAR_ARGS
                                    0132   292
        00    0000'CF   8F         0132   293          CASEB   W^MAC$GL_DIRFLG,#ASC$_ASCIC,- ;DISPATCH
              03                   0137   294                  #ASC$_ASCIZ
                       0009'       0138   295  10$:    .WORD   20$-10$                       ;ASCIC
                       0037'       013A   296          .WORD   30$-10$                       ;ASCID
                       008A'       013C   297          .WORD   40$-10$                       ;ASCII
                       008B'       013E   298          .WORD   50$-10$                       ;ASCIZ
                        05         0140   299          RSB                                   ;NONE OF ABOVE
                                   0141   300
        54    0000'CF   D0         0141   301  20$:    MOVL    W^MAC$GL_ASCCNT,R4            ;ASCIC--GET LENGTH
  000000FF 8F   54      D1         0146   302          CMPL    R4,#255                       ;IS THE STRING TOO LONG?
              08         1F        014D   303          BLSSU   25$                           ;IF LSSU NO
                                   014F   304          $MAC_ERR ASCTOOLONG                   ; Yes--get error code
              FEA9'      30        0154   305          BSBW    MAC$ERRORLN                   ;REPORT THE ERROR
                54       D6        0157   306  25$:    INCL    R4                            ;WHAT WE REALLY WANT IS LENGTH + 1
                72       10        0159   307          BSBB    60$                           ;DO COMMON PART OF PROCESSING
        50    0000'CF   D0         015B   308          MOVL    W^MAC$GL_ASCCNT,R0            ;GET THE CHARACTER COUNT
              FE9D'      30        0160   309          BSBW    MAC$INTOUT_BY                 ;STORE IMMEDIATE BYTE
                                   0163   310          $INTOUT_LW INT$_AUGPC,<W^MAC$GL_ASCCNT> ;RESET PC
                4D       11        016D   311          BRB     35$                           ;AND GO FINISH UP
                                   016F   312
        54    0000'CF   D0         016F   313  30$:    MOVL    W^MAC$GL_ASCCNT,R4           ;ASCID--GET CHARACTER COUNT
  0000FFFF 8F   54      D1         0174   314          CMPL    R4,#^XFFFF                    ;IS THE COUNT TOO LARGE?
              08         1F        017B   315          BLSSU   33$                           ;IF LSSU NO
                                   017D   316          $MAC_ERR ASCTOOLONG                   ; Yes--get error code
              FE7B'      30        0182   317          BSBW    MAC$ERRORLN                   ;REPORT ERROR TO PASS 2
                54  08   C0        0185   318  33$:    ADDL2   #8,R4                         ;WHAT WE NEED IS LENGTH + 8
                43       10        0188   319          BSBB    60$                           ;DO COMMON PROCESSING
                                   018A   320          $INTOUT_LW INT$_SETLONG,<#0,#MAC$GL_LIST_IT> ;DON'T LIST LINE
        54    0000'CF   D0         0198   321          MOVL    W^MAC$GL_ASCCNT,R4            ;GET THE BYTE COUNT
                                   019D   322          $INTOUT_LW INT$_STKL,R4               ;STACK THE LONGWORD
                                   01A5   323          $INTOUT_X INT$_STOW                   ;STORE WORD
        52    0000'CF   06   C1    01AB   324          ADDL3   #6,W^MAC$GL_ASCCNT,R2         ;FIGURE PC ADJUSTMENT
                                   01B1   325          $INTOUT_LW INT$_AUGPC,<R2>            ;RESET PC
                54  08   C0        01B9   326          ADDL2   #8,R4                         ;UPDATE PC ADJUSTMENT
                        05         01BC   327  35$:    $INC_PC R4                            ;RESTORE PC
                                   01C1   328          RSB
                                   01C2   329
                                   01C2   330  40$:                                          ;ASCII
                        05         01C2   331          RSB
                                   01C3   332
                                   01C3   333  50$:                                          ;ASCIZ
```

E 13

CHARACTER STRING ROUTINES   16-SEP-1984 01:58:23   VAX/VMS Macro V04-00   Page 9
CHARACTER DATA GENERATING DIRECTIVES   5-SEP-1984 01:46:46   [MACRO.SRC]ACTCHR.MAR;1   (6)

```
        50.   D4  01C3   334              CLRL    R0                          ;SET TO STORE ZERO BYTE
    FE38'     30  01C5   335              BSBW    MAC$INTOUT_BY               ;STORE IMMEDIATE BYTE
                  01C8   336              $INC_PC                             ;COUNT IT
        05        01CC   337              RSB
                  01CD   338      ;
                  01CD   339      ; COMMON ROUTINE FOR ASCIC AND ASCID
                  01CD   340      ;
                  01CD   341   60$:       $DEC_PC R4                          ;BACK UP PC
    53   54  CE  01D2   342              MNEGL   R4,R3                       ;GET TWO'S COMPLEMENT OF LENGTH
    50   03  9A  01D5   343              MOVZBL  #3,R0                       ;SET TO STORE 3 BYTES
         FE25'  30  01D8   344              BSBW    MAC$INTOUT_N               ;...
    89   22  90  01DB   345              MOVB    #INT$_SETFLAG,(R9)+        ;CODE IS SET FLAG
    89   0C  90  01DE   346              MOVB    #FLGS$V_MEBLST,(R9)+       ;FLAG # TO SET
                  01E1   347              $INTOUT_LW INT$_AUGPC,R3           ;BACKUP PC IN PASS 2
                  01E9   348              $INTOUT_X INT$_FNEWL               ;FORCE NEW LISTING LINE
        05        01EF   349              RSB
```

```
                              01F0   351                    .SBTTL  PACKED DIRECTIVE
                              01F0   352
                              01F0   353          ;++
                              01F0   354          ; FUNCTIONAL DESCRIPTION:
                              01F0   355          ;
                              01F0   356          ;       THIS ROUTINE READS A DECIMAL NUMBER WITH AN OPTIONAL LEADING
                              01F0   357          ;       SIGN.  THE NUMBER IS THEN CONVERTED TO A PACKED DECIMAL STRING
                              01F0   358          ;       AND OUTPUT TO THE INTERMEDIATE CODE AS A STRING OF STORE IMMEDIATE
                              01F0   359          ;       BYTES.
                              01F0   360          ;
                              01F0   361          ;--
                              01F0   362
              FE0D'   30      01F0   363  PACKED::                                    ;DIRECTIVE = KPACKED
        7E    0C      90      01F3   364                    BSBW    MAC$SKIPSP          ;SKIP LEADING SPACES
        2B    5A      91      01F6   365                    MOVB    #12,-(SP)           ;ASSUME POSITIVE NUMBER
              08      13      01F9   366                    CMPB    R10,#^A/+/          ;WAS IT POSITIVE?
        2D    5A      91      01FB   367                    BEQL    10$                 ;IF EQL YES
              06      12      01FE   368                    CMPB    R10,#^A/-/          ;NO--WAS IT NEGATIVE?
        6E    0D      90      0200   369                    BNEQ    20$                 ;NO--NO SIGN AT ALL
              FDFA'   30      0203   370                    MOVB    #13,(SP)            ;YES--SET NEGATIVE SIGN
              56      D4      0206   371  10$:              BSBW    MAC$GETCHR          ;READ CHARACTER AFTER SIGN
  55    0000'CF      9E      0208   372  20$:              CLRL    R6                  ;CLEAR DIGIT COUNTER
                              020D   373                    MOVAB   W^MAC$AB_TMPBUF,R5  ;POINT TO TEMP BUFFER
                              020D   374          ;
                              020D   375          ; HERE FOR EACH NEW CHARACTER
                              020D   376          ;
        2C    5A      91      020D   377  30$:              CMPB    R10,#^A/,/          ;END OF NUMBER?
              4A      13      0210   378                    BEQL    80$                 ;IF EQL YES
        0D    5A      91      0212   379                    CMPB    R10,#CR             ;END OF LINE?
              45      13      0215   380                    BEQL    80$                 ;IF EQL YES
              04      E1      0217   381                    BBC     #CHR$V_NUM BER,-    ;NO--IS CHARACTER A DIGIT?
  1D    0000'CA      0219   382                            W^MAC$AB_CMSK_TAB(R10),50$  ;(BRANCH IF NOT)
        1F    56      D1      021D   383                    CMPL    R6,#31              ;YES--ROOM TO STORE?
              0A      19      0220   384                    BLSS    40$                 ;IF LSS YES
                      0222   385                    $MAC_ERR PACTOOLONG               ; No--string is too long
              FDD6'   30      0227   386                    BSBW    MAC$ERRORLN         ;ISSUE ERROR
              26      11      022A   387                    BRB     70$
              56      D6      022C   388  40$:              INCL    R6                  ;COUNT DIGIT WE STORE
        65    5A      90      022E   389                    MOVB    R10,(R5)            ;STORE DIGIT
  85    F0 8F      8A      0231   390                    BICB2   #^XF0,(R5)+         ;CONVERT TO BINARY DECIMAL
              FDC8'   30      0235   391                    BSBW    MAC$GETCHR          ;GET NEXT CHARACTER
              D3      11      0238   392                    BRB     30$                 ;CONTINUE CHECKING
              FDC5'   30      023A   393  50$:              BSBW    MAC$SKIPSP          ;SKIP SPACES
        2C    5A      91      023D   394                    CMPB    R10,#^A/,/          ;GET TO A COMMA?
              1A      13      0240   395                    BEQL    80$                 ;IF EQL YES
        0D    5A      91      0242   396                    CMPB    R10,#CR             ;NO--EOL?
              15      13      0245   397                    BEQL    80$                 ;IF EQL YES
                      0247   398                    $MAC_ERR NOTDECSTRG               ; No-error
              FDB1'   30      024C   399                    BSBW    MAC$ERRORLN         ;ISSUE ERROR MESSAGE
              FDAE'   30      024F   400  60$:              BSBW    MAC$GETCHR          ;NEXT CHARACTER
        2C    5A      91      0252   401  70$:              CMPB    R10,#^A/,/          ;GET TO COMMA?
              05      13      0255   402                    BEQL    80$                 ;IF EQL YES
        0D    5A      91      0257   403                    CMPB    R10,#CR             ;NO--EOL?
              F3      12      025A   404                    BNEQ    60$                 ;NO--KEEP LOOKING
                              025C   405          ;
                              025C   406          ; HERE WHEN DONE WITH NUMBER
                              025C   407          ;
```

```
          56   D5   025C   408 80$:    TSTL    R6                                      ;NULL STRING?
          04   12   025E   409         BNEQ    90$                                     ;IF NEQ NO
          85   94   0260   410         CLRB    (R5)+                                   ;YES--FAKE 0
          56   D6   0262   411         INCL    R6                                      ;COUNT IT
    85    8E   90   0264   412 90$:    MOVB    (SP)+,(R5)+                             ;STORE SIGN AT END OF STRING
    2C    5A   91   0267   413         CMPB    R10,#^A/,/                              ;COMMA?
          4A   12   026A   414         BNEQ    140$                                    ;IF NEQ NO
          56   DD   026C   415         PUSHL   R6                                      ;YES--SAVE LENGTH
        FD8F'  30   026E   416         BSBW    MAC$GETCHR                              ;GET NEXT CHR
        FD8C'  30   0271   417         BSBW    MAC$SKIPSP                              ;SKIP SPACES
          04   E1   0274   418         BBC     #CHR$V_NUM_BER,-                        ;IS IT A LOCAL LABEL?
 12 0000'CA          0276   419                 W^MAC$AB_CMSK_TAB(R10),100$ ;(BRANCH IF NOT)
        FD83'  30   027A   420         BSBW    MAC$DNUMBER                             ;MAYBE--READ IT
    0C    58   D1   027D   421         CMPL    R8,#ID                                  ;WAS IT AN ID?
    1D    13   0280   422         BEQL    120$                                    ;IF EQL YES--OK
               0282   423         $MAC_ERR DIRSYNX                                ; No--syntax error
        FD76'  30   0287   424         BSBW    MAC$ERRORLN                             ;ISSUE ERROR
          27   11   028A   425         BRB     130$
        FD71'  30   028C   426 100$:   BSBW    MAC$SYMSCNUP                            ;SCAN SYMBOL NAME
    0A    50   E8   028F   427         BLBS    R0,110$                                 ;BRANCH IF GOT ONE
               0292   428         $MAC_ERR DIRSYNX                                ; Syntax error
        FD66'  30   0297   429         BSBW    MAC$ERRORLN
          17   11   029A   430         BRB     130$
        FD61'  30   029C   431 110$:   BSBW    MAC$INSUSRSYMTB                         ;INSERT IN USER SYMTAB
    0111 8F   A8   029F   432 120$:   BISW2   #SYM$M_DEF!SYM$M_ASN!SYM$M_ABS,-  ;SET DEFINED BY ASSIGNMENT
          09   A1   02A3   433                 SYM$W_FLAG(R1)                          ;AND ABSOLUTE
    05 A1  6E   D0   02A5   434         MOVL    (SP),SYM$L_VAL(R1)                      ;SET STRING LENGTH AS VALUE
    55    00'8F 9A   02A9   435         MOVZBL  #CRF$K_DEF,R5                           ;THIS IS A DEFINITION
          56   51   D0   02AD   436         MOVL    R1,R6                                   ;POINT R6 TO SYMBOL BLOCK
        FD4D'  30   02B0   437         BSBW    MAC$CREF_SYM                            ;CREF SYMBOL IF CREFFING
          56 BED0   02B3   438 130$:   POPL    R6                                      ;RESTORE LENGTH
    50    56   02   C1   02B6   439 140$:   ADDL3   #2,R6,R0                                ;ROUND NIBBLES UP TO BYTES
 50    50   FF 8F   78   02BA   440         ASHL    #-1,R0,R0                               ;R0 HAS NUMBER OF BYTES
               02BF   441         $INC_PC R0                                      ;UPDATE PC
    55    0000'CF 9E   02C4   442         MOVAB   W^MAC$AB_TMPBUF,R5                      ;POINT TO TEMP BUFFER
          56   D6   02C9   443         INCL    R6                                      ;COUNT THE SIGN ALSO
    08    56   E9   02CB   444         BLBC    R6,150$                                 ;BRANCH IF EVEN NUMBER OF BYTES
    50    85   90   02CE   445         MOVB    (R5)+,R0                                ;NO--GET FIRST BYTE
        FD2C'  30   02D1   446         BSBW    MAC$INTOUT_BY                           ;EMIT TO PASS 2
          56   D7   02D4   447         DECL    R6                                      ;COUNT IT
          56   D5   02D6   448 150$:   TSTL    R6                                      ;DONE?
          12   15   02D8   449         BLEQ    160$                                    ;IF LEQ YES
    50    85   90   02DA   450         MOVB    (R5)+,R0                                ;NO--GET A NIBBLE
 50    50   04   78   02DD   451         ASHL    #4,R0,R0                                ;MAKE ROOM FOR OTHER NIBBLE
    50    85   88   02E1   452         BISB2   (R5)+,R0                                ;GET SECOND NIBBLE
          56   02   C2   02E4   453         SUBL2   #2,R6                                   ;EAT TWO BYTES
        FD16'  30   02E7   454         BSBW    MAC$INTOUT_BY                           ;EMIT TO INTERM. BUFFER
          EA   11   02EA   455         BRB     150$                                    ;LOOP FOR ALL NIBBLES
          05   02EC   456 160$:   RSB                                             ;DONE
```

```
                    02ED   458              .SBTTL  PC ALIGNMENT DIRECTIVES
                    02ED   459
                    02ED   460   ;++
                    02ED   461   ; FUNCTIONAL DESCRIPTION:
                    02ED   462   ;
                    02ED   463   ;     THESE TWO ROUTINES MAKE THE PC EVEN OR ODD.
                    02ED   464   ;
                    02ED   465   ;--
                    02ED   466
                    02ED   467   EVEN::
06 0000'CF    E8    02ED   468              BLBS    W^MACSGL_PC,ODD_EVEN        ;BRANCH IF PC NEEDS ADJUSTING
              05    02F2   469              RSB                                 ;NO--IT IS ALREADY EVEN
                    02F3   470
                    02F3   471   ODD::
0C 0000'CF    E8    02F3   472              BLBS    W^MACSGL_PC,ODD_EVEN_EXIT  ;BRANCH IF PC IS OK ALREADY
                    02F8   473   ODD_EVEN:
                    02F8   474              $INTOUT_LW INTS_AUGPC,#1            ;AUGMENT PC BY ONE ON PASS 2
                    0300   475              $INC_PC                             ;AUGMENT PC NOW ALSO
                    0304   476   ODD_EVEN_EXIT:
              05    0304   477              RSB
                    0305   478
                    0305   479              .SBTTL  PAGE    MOVE TO NEW LISTING PAGE
                    0305   480
                    0305   481   ;++
                    0305   482   ; FUNCTIONAL DESCRIPTION:
                    0305   483   ;
                    0305   484   ;     THIS ROUTINE EMITS CODE TO PASS 2 TO FORCE A NEW LISTING
                    0305   485   ;     PAGE.
                    0305   486   ;
                    0305   487   ;--
                    0305   488
                    0305   489   PAGE::
                    0305   490              $INTOUT_LW INTS_SETLONG,<#-1,#MACSGL_LINE_CNT> ;FORCE NEW PAGE
                    0317   491              $INTOUT_LW INTS_SETLONG,<#0,#MACSGL_LIST_IT> ;DON'T LIST THE LINE
              05    0325   492              RSB
```

```
                              0326    494                    .SBTTL  ERROR/WARN/PRINT DIRECTIVES
                              0326    495
                              0326    496  ;++
                              0326    497  ; FUNCTIONAL DESCRIPTION:
                              0326    498  ;
                              0326    499  ;      THIS ROUTINE PROCESSES THE .WARN DIRECTIVE.  THE PROPER
                              0326    500  ;      CODES ARE STORED IN 'MAC$GL_VALUE' FOR LATER USE BY
                              0326    501  ;      ERROR2 OR ERROR1.
                              0326    502  ;
                              0326    503  ;--
                              0326    504
                              0326    505  WARN::                                    ;ERROR_HEAD = KWARN
                              0326    506          $MAC_ERR GENWRN                   ; Get message code
                51   38   9A  032B    507          MOVZBL  #INT$_WRN,R1              ;AND INT. BUFFER CODE
                     0F   11  032E    508          BRB     ERR_COM                   ;GO TO COMMON CODE
                              0330    509
                              0330    510  ;++
                              0330    511  ; FUNCTIONAL DESCRIPTION:
                              0330    512  ;
                              0330    513  ;      THIS ROUTINE PROCESSES THE .ERROR DIRECTIVE.  THE PROPER
                              0330    514  ;      CODES ARE STORED IN 'MAC$GL_VALUE' FOR LATER USE BY
                              0330    515  ;      ERROR2 OR ERROR1.
                              0330    516  ;
                              0330    517  ;--
                              0330    518
                              0330    519  ERROR::                                   ;ERROR_HEAD = KERROR
                              0330    520          $MAC_ERR GENERR                   ; Get message code
                51   12   9A  0335    521          MOVZBL  #INT$_ERR,R1              ;AND INT. BUFFER CODE
                     05   11  0338    522          BRB     ERR_COM                   ;GO TO COMMON CODE
                              033A    523
                              033A    524  ;++
                              033A    525  ; FUNCTIONAL DESCRIPTION:
                              033A    526  ;
                              033A    527  ;      THIS ROUTINE PROCESSES THE .PRINT DIRECTIVE.  THE CODES
                              033A    528  ;      ARE SET IN 'MAC$GL_VALUE FOR LATER PROCESSING BY ERROR2
                              033A    529  ;      OR ERROR1.
                              033A    530  ;
                              033A    531  ;--
                              033A    532
                              033A    533  PRINT::                                   ;ERROR_HEAD = KPRINT
                     50   D4  033A    534          CLRL    R0                        ;NO MESSAGE FOR .PRINT
                51   1B   9A  033C    535          MOVZBL  #INT$_PRT,R1              ;SET THE INT. BUFFER CODE
                              033F    536  ;
                              033F    537  ; COMMON ROUTINE FOR ERROR/WARN/PRINT
                              033F    538  ;
                              033F    539  ERR_COM:
          00 6B  06   E5  033F    540          BBCC    #FLG$V_EVALEXPR,(R11),10$ ;DON'T SEND CODE TO PASS 2
       52 0000'CF  9E   0343    541  10$:    MOVAB   W^MAC$GL_VALUE,R2         ;POINT TO RESULT WORD
          82 51    B0   0348    542          MOVW    R1,(R2)+                  ;STORE THE INT. BUFFER CODE
          82 50    B0   034B    543          MOVW    R0,(R2)+                  ;STORE THE MESSAGE INDEX
                              034E    544  ;
                              034E    545  ; COPY LINE FOR PASS 2 ERROR HANDLING
                              034E    546  ;
       56 0000'CF  D0   034E    547  20$:    MOVL    W^MAC$GL_LINELN,R6        ;GET LENGTH OF LINE
       50 56    04  C1   0353    548          ADDL3   #4,R6,R0                  ;FIGURE SIZE OF RECORD
                FCA6'  30   0357    549          BSBW    MAC$INTOUT_N              ;MAKE ROOM FOR IT
 FF A9  FF 8F   90   035A    550          MOVB    #-1,-1(R9)                ;SIGNAL SPECIAL LINE
```

```
                89  13  90  035F  551        MOVB    #INT$_ETX,(R9)+            ;STORE COMMAND
                89      B0  0362  552        MOVW    R6,(R9)+                  ;STORE LENGTH OF LINE
    69  0000'CF 56      28  0365  553        MOVC3   R6,W^MAC$AB_LINEBF,(R9)   ;COPY TEXT INTO BUFFER
                59  53  D0  036B  554        MOVL    R3,R9                     ;UPDATE POINTER
                    05  036E  555            RSB                               ;ALL DONE
```

```
                         036F      557  ;++
                         036F      558  ; FUNCTIONAL DESCRIPTION:
                         036F      559  ;
                         036F      560  ;       ERROR2 AND ERROR1 ARE CALLED AFTER A .ERROR/.WARN/.PRINT
                         036F      561  ;       DIRECTIVE HAS BEEN PROCESSED.  THE INTERMEDIATE CODE AND
                         036F      562  ;       THE MESSAGE INDEX ARE PICKED UP FROM THE STACK, AND EMITTED
                         036F      563  ;       TO THE INTERMEDIATE BUFFER, ALONG WITH THE INPUT LINE BUFFER
                         036F      564  ;       POINTER.  ERROR2 IS CALLED IF THERE WAS AN EXPRESSION TO
                         036F      565  ;       BE PRINTED.  ERROR1 IS CALLED IF THERE WAS NO EXPRESSION.
                         036F      566  ;
                         036F      567  ; NOTE: THE ERROR ROUTINES EXPECT THAT PRIL WILL LEAVE THE EXPR
                         036F      568  ;       SET IN MAC$GL_VALUE.
                         036F      569  ;
                         036F      570  ;
                         036F      571  ;--
                         036F      572
                         036F      573  ERROR2::                             ;DIRECTIVE = ERROR HEAD EXPR
                         036F      574          $INTOUT_LW INT$_PRIL,<W^MAC$AL_VALSTACK[R7]> ;PRINT VALUE
      52    FFFC'CF47    DE  037A   575          MOVAL   W^MAC$AL_VALSTACK-4[R7],R2 ;POINT TO WHERE THE DATA IS
            14    11    0380   576          BRB     ERR_3                    ;GO FINISH UP
                        0382   577
                        0382   578  ERROR1::                             ;DIRECTIVE = ERROR HEAD
                        0382   579          $INTOUT_LW INT$_SETLONG,<#0,#MAC$GL_VALUE> ;CLEAR VALUE RESIDUE IN PASS 2
      52    0000'CF47    DE  0390   580          MOVAL   W^MAC$AL_VALSTACK[R7],R2 ;POINT TO WHERE THE DATA IS
            50    0A    9A  0396   581  ERR_3:  MOVZBL  #10,R0                   ;THERE WILL BE 10 BYTES
            FC64'      30  0399   582          BSBW    MAC$INTOUT_N             ;SET UP FOR IT
      89    82    90  039C   583          MOVB    (R2)+,(R9)+              ;SET THE COMMAND FOR PASS 2
            52    D6  039F   584          INCL    R2                       ;SKIP A BYTE
      89    82    3C  03A1   585          MOVZWL  (R2)+,(R9)+              ;FIRST LONGWORD IS MESSAGE INDEX
  89  0000'CF    D0  03A4   586          MOVL    W^MAC$GL_LINEPT,(R9)+     ;SECOND LONGWORD IS LINE POINTER
      00 6B    06    E3  03A9   587          BBCS    #FLGSV_EVALEXPR,(R11),10$ ;ALLOW EXPRESSION EVALUATION
            05  03AD   588  10$:    RSB
```

```
                         03AE    590                 .SBTTL  NCHR    NUMBER OF CHARACTERS DIRECTIVE
                         03AE    591
                         03AE    592         ;++
                         03AE    593         ; FUNCTIONAL DESCRIPTION:
                         03AE    594         ;
                         03AE    595         ;       THIS DIRECTIVE DEFINES THE SYMBOL SUPPLIED WITH THE LENGTH
                         03AE    596         ;       OF THE FOLLOWING STRING.
                         03AE    597         ;
                         03AE    598         ;--
                         03AE    599
                         03AE    600 NCHR::                                              ;DIRECTIVE = KNCHR ID
           FC4F'  30     03AE    601                 BSBW    MAC$SKIPSP                  ;SKIP SPACES
      2C   5A    91      03B1    602                 CMPB    R10,#^A/,/                  ;COMMA?
           06    12      03B4    603                 BNEQ    10$                         ;IF NEQ NO
           FC47'  30     03B6    604                 BSBW    MAC$GETCHR                  ;YES--SKIP IT
           FC44'  30     03B9    605                 BSBW    MAC$SKIPSP                  ;SKIP SPACES
           FC41'  30     03BC    606 10$:            BSBW    MAC$MAC_ARG_SCN             ;SCAN THE ARGUMENT
 56   0000'CF47  D0      03BF    607                 MOVL    W^MAC$AL_VA[STACK[R7],R6    ;POINT TO SYMBOL BLOCK
           00AB   31     03C5    608                 BRW     FINISH_N_DIR               ;FINISH PROCESSING DIRECTIVE
```

MAC$ACTCHR
V04-000

M 13

CHARACTER STRING ROUTINES   16-SEP-1984 01:58:23 VAX/VMS Macro V04-00   Page 17
NARG RETURN NUMBER OF ARGUMENTS IN MACRO   5-SEP-1984 01:46:46 [MACRO.SRC]ACTCHR.MAR;1   (12)

```
                          03C8   610              .SBTTL  NARG    RETURN NUMBER OF ARGUMENTS IN MACRO CALL
                          03C8   611
                          03C8   612    ;++
                          03C8   613    ; FUNCTIONAL DESCRIPTION:
                          03C8   614    ;
                          03C8   615    ;       THIS DIRECTIVE DEFINES THE SYMBOL SUPPLIED TO BE THE NUMBER
                          03C8   616    ;       OF ARGUMENTS IN THE CURRENT MACRO CALL.  AN ERROR IS GENERATED
                          03C8   617    ;       IF WE ARE NOT CURRENTLY EXPANDING A MACRO.
                          03C8   618    ;
                          03C8   619    ;--
                          03C8   620
                          03C8   621    NARG::                                        ;DIRECTIVE = KNARG
              FC35'  30   03C8   622              BSBW    MAC$SYMSCNUP                 ;SCAN THE ID
           08 50   E8     03CB   623              BLBS    R0,10$                       ;BRANCH IF ID THERE
                          03CE   624              $MAC_ERR DIRSYNX                     ; No--syntax error
              FC2A'  31   03D3   625              BRW     MAC$ERRORPT                  ;ISSUE ERROR AND RETURN
              FC27'  30   03D6   626    10$:      BSBW    MAC$INSUSRSYMTB              ;INSERT IN USER SYMBOL TABLE
     55  0000'CF   D0     03D9   627              MOVL    W^MAC$GL_INPUTP,R5           ;POINT TO CURRENT INPUT BLOCK
        56    51   D0     03DE   628              MOVL    R1,R6                        ;POINT TO SYMBOL BLOCK
     08 6B   10    E0     03E1   629              BBS     #FLG$V_MACTXT,(R11),20$      ;BRANCH IF READING MACRO TEXT
                          03E5   630              $MAC_ERR NOTINMACRO                  ; No--not in macro
              FC13'  30   03EA   631              BSBW    MAC$ERRORPT                  ;ISSUE ERROR
     50  1C A5   9A       03ED   632    20$:      MOVZBL  INP$B_ARGCT(R5),R0          ;GET THE NUMBER OF ARGS
           007F   31      03F1   633              BRW     FINISH_N_DIR                 ;GO FINISH PROCESSING DIRECITVE
```

```
                              03F4    635                 .SBTTL  NTYPE    ASSIGN OPERAND TYPE TO SYMBOL
                              03F4    636
                              03F4    637         ;++
                              03F4    638         ; FUNCTIONAL DESCRIPTION:
                              03F4    639         ;
                              03F4    640         ;--
                              03F4    641
                              03F4    642         NTHD2::                                         ;NTYPE HEAD = KNTYPE ID
        50    0000'CF47   DO  03F4    643                 MOVL    W^MAC$AL_VALSTACK[R7],R0 ;GET ID BLOCK ADDRESS
                   06     11  03FA    644                 BRB     NTYP_0                  ;JOIN COMMON CODE
                              03FC    645
                              03FC    646         NTHD1::                                         ;NTYPE HEAD = KNTYPE ID DCOMMA
        50  FFFC'CF47     DO  03FC    647                 MOVL    W^MAC$AL_VALSTACK-4[R7],R0 ;GET ID BLOCK ADDRESS
        0000'CF    50     DO  0402    648         NTYP_0: MOVL    R0,W^MAC$GL_ASNPTR        ;SAVE ID BLOCK ADDRESS
   0000'CF    0000'CF     DO  0407    649                 MOVL    W^MAC$GL_PC,W^MAC$GL_SAVE_PC ;SAVE PC ADDRESS
        0000'CF           D4  040E    650                 CLRL    W^MAC$GL_ABSFLAG          ;ASSUME ABSOLUTE EXPRESSION
        0000'CF    59     D1  0412    651                 CMPL    R9,W^MAC$GL_INTWRNPT      ;TIME TO FLUSH BUFFER?
                   03     1B  0417    652                 BLEQU   20$                      ;IF LEQ NO
              FBE4'        30  0419    653                 BSBW    MAC$OUTFRAME             ;YES--FLUSH THE BUFFER
        0000'CF    59     DO  041C    654         20$:    MOVL    R9,W^MAC$GL_EXPPTR        ;SET EXPRESSION POINTER
        0000'CF    59     DO  0421    655                 MOVL    R9,W^MAC$GL_EXPEND        ;AND END OF EXPRESSION
           00 6B    06    E5  0426    656                 BBCC    #FLG$V_EVALEXPR,(R11),.+1 ;DO NOT EVALUATE EXPRESSION
     6B  01000084 8F      C8  042A    657                 BISL2   #FLG$M_COMPEXPR!FLG$M_EXPOPT!FLG$M_NOREF,(R11) ;ASSUME COMPILE-TIME
                              0431    658                                                  ;AND ALLOW OPTIMIZATION
                              0431    659                                                  ;TELL PRMSYM NOT TO SET REF BIT
        0000'CF    10     DO  0431    660                 MOVL    #184,W^MAC$GL_OPSIZE     ;SET READ ACCESS MODE
           00 6B    25    E2  0436    661                 BBSS    #FLG$V_NTYPEPC,(R11),.+1 ; Suppress normal PC register checks
                   05         043A    662                 RSB
                              043B    663
                              043B    664         ;++
                              043B    665         ; FUNCTIONAL DESCRIPTION:
                              043B    666         ;
                              043B    667         ;       THESE TWO ROUTINES PERFORM THE ACTUAL ASSIGNMENT OF THE
                              043B    668         ;       MODE/REGISTER TO THE SYMBOL POINTED TO BY MAC$GL_ASNPTR.
                              043B    669         ;
                              043B    670         ;--
                              043B    671
                              043B    672         NTYPE1::                                        ;DIRECTIVE = NTYPE_HEAD
                              043B    673                 $MAC_ERR DIRSYNX                 ; No REF!! error.
              FBBD'        30  0440    674                 BSBW    MAC$ERRORLN             ;SO ISSUE A
        0000'CF           D4  0443    675                 CLRL    W^MAC$GL_VALUE           ;CLEAR RESULT
                              0447    676
                              0447    677         NTYPE2::                                        ;DIRECTIVE = NTYPE_HEAD REF
   0000'CF    G000'CF     DO  0447    678                 MOVL    W^MAC$GL_SAVE_PC,W^MAC$GL_PC ;RESET PC
        56    0000'CF     DO  044E    679                 MOVL    W^MAC$GL_ASNPTR,R6       ;POINT TO THE SYMBOL BLOCK
        50    0000'CF     9A  0453    680                 MOVZBL  W^MAC$GB_IMODE,R0        ;GET IMODE
        50    50    04    78  0458    681                 ASHL    #4,R0,R0                 ;MAKE ROOM FOR IREG
        50    0000'CF     88  045C    682                 BISB2   W^MAC$GB_IREG,R0         ;ADD IN IREG
        50    50    04    78  0461    683                 ASHL    #4,R0,R0                 ;MAKE ROOM FOR MODE
        50    0000'CF     88  0465    684                 BISB2   W^MAC$GB_MODE,R0         ;ADD IN MODE
        50    50    04    78  046A    685                 ASHL    #4,R0,R0                 ;MAKE ROOM FOR REG
        50    0000'CF     88  046E    686                 BISB2   W^MAC$GB_REG,R0          ;THE TYPE IS COMPLETE
                              0473    687         FINISH_N_DIR:
        0000'CF           D4  0473    688                 CLRL    W^MAC$GB_MODE            ;CLEAR REGS AND MODE
        05 A6    50       DO  0477    689                 MOVL    R0,SYM$L_VAL(R6)         ;STORE IN SYMBOL BLOCK
        OC A6            94  047B    690                 CLRB    SYM$B_SEG(R6)            ;DEFINE IN ABSOLUTE PSECT
              FB7F'        30  047E    691                 BSBW    MAC$MOL_DEF_CHK         ;CHECK FOR MULTIPLY DEFINED
```

MAC$ACTCHR   CHARACTER STRING ROUTINES   16-SEP-1984 01:58:23 VAX/VMS Macro V04-00   Page 19
V04-000   NTYPE ASSIGN OPERAND TYPE TO SYMBOL   5-SEP-1984 01:46:46 LMACRO.SRCJACTCHR.MAR;1   (13)

B 14

```
        0111 8F   A8  0481   692      BISW2   #SYM$M_DEF!SYM$M_ABS!SYM$M_ASN,- ;SET SYMBOL FLAGS
           09.A6       0485   693              SYM$W_FLAG(R6)   ;
55   00'8F   9A  0487   694      MOVZBL  #CRF$K_DEF,R5           ;THIS IS A DEFINITION
        FB72'    30  048B   695      BSBW    MAC$CREF_SYM           ;CREF SYMBOL IF CREFFING
        FB6F'    30  048E   696      BSBW    MAC$INTOUT_ASN         ;OUTPUT ASN TO PASS 2
                     0491   697      $INTOUT_LW INT$_PRIL,<SYM$L_VAL(R6)> ;PRINT VALUE
           05       049A   698      RSB
                    049B   699
                    049B   700      .END
```

```
$COUNT              = 0000003B              DPOUND             = 00000021
ARG$K_SIZE          = 000003E8              DSC$K_DTYPE_T      = 0000000E
ASC$_XSCIC          = 00000000              DSQCLS             = 00000014
ASC$_ASCID          = 00000001              DSQOPN             = 00000013
ASC$_ASCII          = 00000002              DSUP               = 0000002F
ASC$_ASCIZ          = 00000003              DTIMES             = 0000001B
ASCIC                 00000000 RG    03     DUPA               = 00000023
ASCID                 00000015 RG    03     DUPB               = 00000024
ASCII                 00000052 RG    03     DUPC               = 00000025
ASCIZ                 00000055 RG    03     DUPD               = 00000026
ASC_COM               00000058 R     03     DUPF               = 00000028
AUD$K_SIZE          = 00000010              DUPM               = 00000029
BLNK                = 00000020              DUPO               = 00000027
CHARHD                00000132 RG    03     DUPX               = 0000002A
CHR$M_COMMA_CR      = 00000020              DWUP               = 00000030
CHR$M_ILL_CHR       = 00000040              DXOR               = 0000001F
CHR$M_NUM_BER       = 00000010              ERR                = 00000000
CHR$M_SPA_MSK       = 00000001              ERR01              = 00000001
CHR$M_SYM_CH1       = 00000008              ERR02              = 00000002
CHR$M_SYM_CHR       = 00000004              ERR03              = 00000003
CHR$M_SYM_DLM       = 00000002              ERR04              = 00000004
CHR$V_COMMA_CR      = 00000005              ERR05              = 00000005
CHR$V_CVTLWC        = 00000061              ERR06              = 00000006
CHR$V_ILL_CHR       = 00000006              ERR07              = 00000007
CHR$V_NOCVT         = 0000007F              ERR08              = 00000008
CHR$V_NUM_BER       = 00000004              ERR09              = 00000009
CHR$V_SPA_MSK       = 00000000              ERROR                00000330 RG    03
CHR$V_SYM_CH1       = 00000003              ERROR1               00000382 RG    03
CHR$V_SYM_CHR       = 00000002              ERROR2               0000036F RG    03
CHR$V_SYM_DLM       = 00000001              ERR_3                00000396 R     03
CHRARG                0000008A RG    03     ERR_COM              0000033F R     03
CHRNUL                0000007B RG    03     EVEN                 000002ED RG    03
CNT                 = 00000001              FF                 = 0000000C
CR                  = 0000000D              FINISH_N_DIR         00000473 R     03
CRF$K_DEF             ******** X     03     FLG$M_ALCCHR       = 00000001
DAND                = 0000001D              FLG$M_BOL          = 00000002
DANGCLS             = 00000016              FLG$M_CHKLPND      = 00100000
DANGOPN             = 00000015              FLG$M_COMPEXPR     = 00000004
DAT                 = 00000020              FLG$M_CONT         = 00000008
DBUP                = 0000002B              FLG$M_CRF          = 40000000
DCLS                = 00000018              FLG$M_CRSEEN       = 00000001
DCOLON              = 00000010              FLG$M_DATRPT       = 00000010
DCOMMA              = 0000000F              FLG$M_DBGOUT       = 00004000
DDIV                = 0000001C              FLG$M_DLIMSTR      = 00008000
DEOL                = 0000000B              FLG$M_ENDMCH       = 00000020
DEQ                 = 00000011              FLG$M_EVALEXPR     = 00000040
DGUP                = 0000002C              FLG$M_EXPOPT       = 00000080
DINTEGER            = 00000022              FLG$M_EXTERR       = 00010000
DIUP                = 0000002D              FLG$M_EXTWRN       = 00020000
DLUP                = 0000002E              FLG$M_FIRSTLN      = 00000200
DMASK               = 00000032              FLG$M_IFSTAT       = 00800000
DMINUS              = 0000001A              FLG$M_IIF          = 00400000
DOPCODE             = 0000000E              FLG$M_INSERT       = 00000100
DOPN                = 00000017              FLG$M_IRPC         = 20000000
DOR                 = 0000001E              FLG$M_LEXOP        = 00000002
DPC                 = 00000012              FLG$M_LSTXST       = 00000200
DPLUS               = 00000019              FLG$M_MAC2COL      = 00000800
```

MAC$ACTCHR
Symbol table

D 14

CHARACTER STRING ROUTINES

16-SEP-1984 01:58:23   VAX/VMS Macro V04-00   Page 21
5-SEP-1984 01:46:46   [MACRO.SRC]ACTCHR.MAR;1   (13)

FLG$M_MACL	= 00000800	FLG$V_NEWPND	= 0000000A	
FLG$M_MACLTB	= 08000000	FLG$V_NOREF	= 00000018	
FLG$M_MACTXT	= 00010000	FLG$V_NTYPEPC	= 00000025	
FLG$M_MEBLST	= 00001000	FLG$V_NULCHR	= 00000032	
FLG$M_MOREARG	= 00002000	FLG$V_OBJXST	= 00000015	
FLG$M_MOREINP	= 00000008	FLG$V_OPNDCHK	= 00000028	
FLG$M_NEWPND	= 00000400	FLG$V_OPRND	= 0000000D	
FLG$M_NOREF	= 01000000	FLG$V_OPTVFLIDX	= 0000002C	
FLG$M_NTYPEPC	= 00000020	FLG$V_ORDLST	= 00000011	
FLG$M_NULCHR	= 00040000	FLG$V_P2	= 0000000E	
FLG$M_OBJXST	= 00200000	FLG$V_RPTIRP	= 0000001C	
FLG$M_OPNDCHK	= 00000100	FLG$V_SEQFIL	= 00000019	
FLG$M_OPRND	= 00002000	FLG$V_SKAN	= 0000000F	
FLG$M_OPTVFLIDX	= 00001000	FLG$V_SPECOP	= 00000022	
FLG$M_ORDLST	= 00020000	FLG$V_SPLALL	= 0000001A	
FLG$M_P2	= 00004000	FLG$V_STOIMF	= 00000012	
FLG$M_RPTIRP	= 10000000	FLG$V_SYM2COL	= 0000002A	
FLG$M_SEQFIL	= 02000000	FLG$V_TOCFLG	= 00000013	
FLG$M_SKAN	= 00008000	FLG$V_UPAFLG	= 00000024	
FLG$M_SPECOP	= 00000004	FLG$V_UPDFIL	= 00000027	
FLG$M_SPLALL	= 04000000	FLG$V_UPMARG	= 00000026	
FLG$M_STOIMF	= 00040000	FLG$V_XCRF	= 0000001F	
FLG$M_SYM2COL	= 00000400	GET_CHARS	000000B4 R	03
FLG$M_TOCFLG	= 00080000	GOAL$Y	= 0000000A	
FLG$M_UPAFLG	= 00000010	HASHSZ	= 0000007F	
FLG$M_UPDFIL	= 00000080	HYPHEN	= 0000002D	
FLG$M_UPMARG	= 00000040	ID	= 0000000C	
FLG$M_XCRF	= 80000000	INP$B_ARGCT	0000001C	
FLG$V_ALLCHR	= 00000000	INP$K_BLKSIZ	00000021	
FLG$V_BOL	= 00000001	INP$K_BUFSIZ	= 000003E8	
FLG$V_CHKLPND	= 00000014	INP$K_IRPSIZ	0000003C	
FLG$V_COMPEXPR	= 00000002	INP$L_ARGS	0000001D	
FLG$V_CONT	= 00000003	INP$L_GETL	00000008	
FLG$V_CRF	= 0000001E	INP$L_IFLVL	0000000C	
FLG$V_CRSEEN	= 00000020	INP$L_IFVAL	00000010	
FLG$V_DATRPT	= 00000004	INP$L_LINK	00000000	
FLG$V_DBGOUT	= 0000002E	INP$L_NXTL	00000004	
FLG$V_DLIMSTR	= 0000002F	INP$L_PAGP	00000018	
FLG$V_ENDMCH	= 00000005	INP$L_RPTCNT	00000014	
FLG$V_EVALEXPR	= 00000006	INT$K_BUFSIZ	= 000013F4	
FLG$V_EXPOPT	= 00000007	INT$K_BUFWRN	= 00001390	
FLG$V_EXTERR	= 00000030	INT$_ADD	= 00000001	
FLG$V_EXTWRN	= 00000031	INT$_AND	= 00000002	
FLG$V_FIRSTLN	= 00000029	INT$_ASH	= 00000003	
FLG$V_IFSTAT	= 00000017	INT$_ASN	= 0000000C	
FLG$V_IIF	= 00000016	INT$_AUGPC	= 0000000D	
FLG$V_INSERT	= 00000008	INT$_BDST	= 0000000E	
FLG$V_IRPC	= 0000001D	INT$_CHKL	= 0000000F	
FLG$V_LEXOP	= 00000021	INT$_DIV	= 00000004	
FLG$V_LSTXST	= 00000009	INT$_END	= 00000010	
FLG$V_MAC2COL	= 0000002B	INT$_EPT	= 00000011	
FLG$V_MACL	= 0000000B	INT$_ERR	= 00000012	
FLG$V_MACLTB	= 0000001B	INT$_ETX	= 00000013	
FLG$V_MACTXT	= 00000010	INT$_FNEWL	= 00000014	
FLG$V_MEBLST	= 0000000C	INT$_ILG	= 00000000	
FLG$V_MOREARG	= 0000002D	INT$_INFO	= 0000003A	
FLG$V_MOREINP	= 00000023	INT$_LGLAB	= 00000015	

Symbol	Value	Symbol	Value
INT$_MACL	= 00000016	KBLKQ	= 00000044
INT$_MUL	= 00000005	KBLKW	= 00000045
INT$_NEG	= 00000006	KBYTE	= 00000038
INT$_NEWL	= 00000017	KCROSS	= 00000079
INT$_NEWP	= 00000018	KDEBUG	= 00000055
INT$_NOT	= 00000007	KDFLT	= 0000007B
INT$_OP	= 00000019	KDOUBLE	= 00000039
INT$_OR	= 00000008	KDSABL	= 00000056
INT$_PRIL	= 0000001A	KENABL	= 00000057
INT$_PRT	= 0000001B	KEND	= 00000076
INT$_PSECT	= 0000001C	KENDC	= 0000004E
INT$_REDEF	= 0000001D	KENDM	= 00000053
INT$_REF	= 0000001E	KENDR	= 0000004F
INT$_REST	= 0000001F	KENTRY	= 00000058
INT$_SAME	= 00000009	KERROR	= 00000071
INT$_SAVE	= 00000020	KEVEN	= 0000005B
INT$_SBTTL	= 00000021	KEXTRN	= 0000005D
INT$_SETFLAG	= 00000022	KFIELD	= 0000003A
INT$_SETLONG	= 00000023	KFLOAT	= 0000003B
INT$_SPIC	= 00000024	KGFLOAT	= 00000081
INT$_SPID	= 00000025	KGLOBL	= 0000005E
INT$_STIB	= 00000026	KHFLOAT	= 00000082
INT$_STIL	= 00000028	KIDENT	= 0000006A
INT$_STIW	= 00000027	KIF	= 00000046
INT$_STKEPT	= 00000029	KIFF	= 00000048
INT$_STKG	= 0000002A	KIFT	= 00000049
INT$_STKL	= 0000002B	KIFTF	= 0000004A
INT$_STKPC	= 0000002C	KIIF	= 00000047
INT$_STKS	= 0000002D	KINCLUDE	= 0000005F
INT$_STOB	= 00000034	KIRP	= 0000004B
INT$_STOL	= 0000002E	KIRPC	= 0000004C
INT$_STOW	= 00000035	KLIBRARY	= 00000060
INT$_STRB	= 0000002F	KLINK	= 00000085
INT$_STRL	= 00000031	KLIST	= 00000061
INT$_STRSB	= 00000032	KLONG	= 0000003C
INT$_STRSW	= 00000033	KMACRO	= 00000050
INT$_STRW	= 00000030	KMCALL	= 00000051
INT$_STSB	= 00000036	KMDELETE	= 00000054
INT$_STSW	= 00000037	KMEXIT	= 00000052
INT$_SUB	= 0000000A	KNARG	= 00000063
INT$_SUME	= 00000039	KNCHR	= 00000064
INT$_WRN	= 00000038	KNCROS	= 0000007A
INT$_XOR	= 0000000B	KNLIST	= 00000062
KADDRESS	= 00000037	KNTYPE	= 00000074
KALIGN	= 0000005A	KOCTA	= 00000083
KASCIC	= 00000033	KODD	= 0000005C
KASCID	= 00000078	KOPDEF	= 00000075
KASCII	= 00000034	KPACKED	= 00000036
KASCIZ	= 00000035	KPAGE	= 00000065
KBLKA	= 0000003F	KPRINT	= 00000072
KBLKB	= 00000040	KPSECT	= 00000066
KBLKD	= 00000041	KQUAD	= 0000003D
KBLKF	= 00000042	KREF1	= 0000006D
KBLKG	= 0000007E	KREF16	= 00000084
KBLKH	= 0000007F	KREF2	= 0000006E
KBLKL	= 00000043	KREF4	= 0000006F
KBLKO	= 00000080	KREF8	= 00000070

```
KREPT                = 0000004D          MAC$INTOUT_X                 ********  X   03
KRESTORE             = 00000067          MAC$MAC_ARG_SCN              ********  X   03
KSAVE                = 00000068          MAC$MUL_DEF_CHK              ********  X   03
KSBTTL               = 0000006B          MAC$OPTIMIZEXPR             ********  X   03
KSGNB                = 0000007C          MAC$OUTFRAME                 ********  X   03
KSGNW                = 0000007D          MAC$SKIPSP                   ********  X   03
KTITLE               = 00000069          MAC$SYMSCNUP                 ********  X   03
KVECTOR              = 00000059          MAC$_ASCTOOLONG          = 007D901A
KWARN                = 00000073          MAC$_DIRSYNX             = 007D906A
KWEAK                = 0000006C          MAC$_GENERR              = 007D908A
KWORD                = 0000003E          MAC$_GENWRN              = 007D8818
KXFER                = 00000077          MAC$_NOTDECSTRG          = 007D916A
LST$K_BUFSIZ         = 00000086          MAC$_NOTINMACRO          = 007D918A
LST$K_L_P_PAGE       = 0000003C          MAC$_PACTOOLONG          = 007D91AA
LST$K_TITLE_SIZ      = 00000028          MAC$_UNTERMARG           = 007D922A
MAB$B_ARGNO            00000005          MACTXT                   = 0000000D
MAB$B_NAME             00000004          MAC_SUBSYS               = 0000007D
MAB$K_BLKSIZ           0000000C          MNB$B_ARGCT                00000017
MAB$L_DVPTR            00000008          MNB$B_NAME                 00000004
MAB$L_LINK             00000000          MNB$K_BLKSIZ               0000001C
MAB$W_DVLEN            00000006          MNB$L_ARGP                 00000018
MAC$AB_CMSK_TAB       ********  X   03   MNB$L_CRSYMF               00000013
MAC$AB_LINEBF         ********  X   03   MNB$L_LINK                 00000000
MAC$AB_TMPBUF         ********  X   03   MNB$L_PAGC                 0000000F
MAC$AL_VALSTACK       ********  X   03   MNB$L_PAGP                 0000000B
MAC$CREF_SYM          ********  X   03   MNB$L_TXTP                 00000005
MAC$DNUMBER           ********  X   03   MNB$W_FLAG                 00000009
MAC$ERRORLN           ********  X   03   MXB$K_BLKSIZ               00000008
MAC$ERRORPT           ********  X   03   MXB$L_LINK                 00000000
MAC$GB_IMODE          ********  X   03   MXB$L_PAGES                00000004
MAC$GB_IREG           ********  X   03   NARG                       000003C8  RG   03
MAC$GB_MODE           ********  X   03   NCHR                       000003AE  RG   03
MAC$GB_REG            ********  X   03   NTHD1                      000003FC  RG   03
MAC$GETCHR            ********  X   03   NTHD2                      000003F4  RG   03
MAC$GL_ABSFLAG        ********  X   03   NTYPE1                     0000043B  RG   03
MAC$GL_ASCCNT         ********  X   03   NTYPE2                     00000447  RG   03
MAC$GL_ASNPTR         ********  X   03   NTYP_0                     00000402  R    03
MAC$GL_DIRFLG         ********  X   03   OBJ$R_BUFSIZ             = 00000200
MAC$GL_EXPEND         ********  X   03   ODD                        000002F3  RG   03
MAC$GL_EXPPTR         ********  X   03   ODD_EVEN                   000002F8  R    03
MAC$GL_INPUTP         ********  X   03   ODD_EVEN_EXIT              00000304  R    03
MAC$GL_INTWRNPT       ********  X   03   OPF$M_LASTOPR           = 00002000
MAC$GL_LINELN         ********  X   03   OPF$M_OPTEXP            = 00001000
MAC$GL_LINEPT         ********  X   03   OPF$V_LASTOPR           = 0000000D
MAC$GL_LINE_CNT       ********  X   03   OPF$V_OPTEXP            = 0000000C
MAC$GL_LIST_IT        ********  X   03   PACKED                     000001F0  RG   03
MAC$GL_OPSIZE         ********  X   03   PAGE                       00000305  RG   03
MAC$GL_PC             ********  X   03   PRINT                      0000033A  RG   03
MAC$GL_PRMSEG         ********  X   03   PSC$B_NAME                 00000004
MAC$GL_SAVE_PC        ********  X   03   PSC$B_SEG                  0000000C
MAC$GL_VALUE          ********  X   03   PSC$B_UNUSED               0000000B
MAC$INSUSRSYMTB       ********  X   03   PSC$K_BLKSIZ               00000013
MAC$INTOUT_1_LW       ********  X   03   PSC$K_NO_OPTNS          = 0000000A
MAC$INTOUT_2_LW       ********  X   03   PSC$L_CURLOC               0000000F
MAC$INTOUT_ASN        ********  X   03   PSC$L_LINK                 00000000
MAC$INTOUT_BY         ********  X   03   PSC$L_MAXLGTH              00000005
MAC$INTOUT_N          ********  X   03   PSC$M_ABS               = FFFFFFF7
```

PSC$M_ALIGNFLG	= 00004000	SYM$K_BLKSIZ	0000000D	
PSC$M_ALLOPTNS	= 000003FF	SYM$K_MAXLEN	= 0000001F	
PSC$M_BYTE	= 00004000	SYM$K_TWOCOL	= 00000010	
PSC$M_CON	= FFFFFFFB	SYM$L_LINK	00000000	
PSC$M_DEFAULT	= 000001C8	SYM$L_VAL	00000005	
PSC$M_EXE	= 000000C0	SYM$M_ABS	= 00000010	
PSC$M_GBL	= 00000010	SYM$M_ASN	= 00000100	
PSC$M_LCL	= FFFFFFEF	SYM$M_CRFO	= 00002000	
PSC$M_LIB	= 00000002	SYM$M_DEBUG	= 00000020	
PSC$M_LONG	= 00004800	SYM$M_DEF	= 00000001	
PSC$M_NOEXE	= FFFFFFBF	SYM$M_DELMAC	= 00000200	
PSC$M_NOPIC	= FFFFFFFE	SYM$M_EPT	= 00000200	
PSC$M_NORD	= FFFFFF7F	SYM$M_EXTRN	= 00000008	
PSC$M_NOSHR	= FFFFFFDF	SYM$M_GLOBL	= 00000004	
PSC$M_NOVEC	= FFFFFDFF	SYM$M_LOCAL	= 00000040	
PSC$M_NOWRT	= FFFFFEFF	SYM$M_ODBG	= 00000400	
PSC$M_OVR	= 00000004	SYM$M_REF	= 00000080	
PSC$M_PAGE	= 00006400	SYM$M_RELPSECT	= 00000800	
PSC$M_PIC	= 00000001	SYM$M_SUPR	= 00004000	
PSC$M_QUAD	= 00004C00	SYM$M_WEAK	= 00000002	
PSC$M_RD	= 00000080	SYM$M_XCRF	= 00001000	
PSC$M_REL	= 00000008	SYM$V_ABS	= 00000004	
PSC$M_SHR	= 00000020	SYM$V_ASN	= 00000008	
PSC$M_USR	= FFFFFFFD	SYM$V_CRFO	= 0000000D	
PSC$M_VEC	= 00000200	SYM$V_DEBUG	= 00000005	
PSC$M_WORD	= 00004400	SYM$V_DEF	= 00000000	
PSC$M_WRT	= 00000180	SYM$V_DELMAC	= 00000009	
PSC$S_ALIGNMENT	= 00000004	SYM$V_EPT	= 00000009	
PSC$V_ALIGNFLG	= 0000000E	SYM$V_EXTRN	= 00000003	
PSC$V_ALIGNMENT	= 0000000A	SYM$V_GLOBL	= 00000002	
PSC$V_EXE	= 00000006	SYM$V_LOCAL	= 00000006	
PSC$V_GBL	= 00000004	SYM$V_ODBG	= 0000000A	
PSC$V_LIB	= 00000001	SYM$V_REF	= 00000007	
PSC$V_OVR	= 00000002	SYM$V_RELPSECT	= 0000000B	
PSC$V_PIC	= 00000000	SYM$V_SUPR	= 0000000E	
PSC$V_RD	= 00000007	SYM$V_WEAK	= 00000001	
PSC$V_REL	= 00000003	SYM$V_XCRF	= 0000000C	
PSC$V_SHR	= 00000005	SYM$W_FLAG	00000009	
PSC$V_VEC	= 00000009	TAB	= 00000009	
PSC$V_WRT	= 00000008	WARN	00000326 RG 03	
PSC$W_FLAG	00000009	X1	= 00000400	
PSC$W_OPTIONS	0000000D	X2	= 0000000F	
RDX$V_BINARY	= 00000000			
RDX$V_DECIMAL	= 00000002			
RDX$V_DOUBLE	= 00000005			
RDX$V_FLOAT	= 00000004			
RDX$V_GFLOAT	= 00000006			
RDX$V_HEX	= 00000003			
RDX$V_HFLOAT	= 00000007			
RDX$V_OCTAL	= 00000001			
REG$_PC	= 0000000F			
RRREG	= 00000031			
SEMI	= 0000003B			
STB$K_PG_MISS	= 0000000A			
SYM$B_NAME	00000004			
SYM$B_SEG	0000000C			
SYM$B_TOKEN	0000000B			

```
                              +------------------+
                              ! Psect synopsis !
                              +------------------+

PSECT name                    Allocation        PSECT No.  Attributes
----------                    ----------        ---------  ----------
.  ABS  .                     00000000 (     0.)  00 (   0.)  NOPIC   USR   CON   ABS   LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
.  BLANK .                    00000000 (     0.)  01 (   1.)  NOPIC   USR   CON   REL   LCL NOSHR  EXE   RD     WRT NOVEC BYTE
$ABS$                         0000003C (    60.)  02 (   2.)  NOPIC   USR   CON   ABS   LCL NOSHR  EXE   RD     WRT NOVEC BYTE
MAC$RO_CODE_P1                0000049B (  1179.)  03 (   3.)  NOPIC   USR   CON   REL   GBL NOSHR  EXE   RD   NOWRT NOVEC LONG

                              +----------------------------+
                              ! Performance indicators !
                              +----------------------------+

Phase                 Page faults     CPU Time        Elapsed Time
-----                 -----------     --------        ------------
Initialization              29      00:00:00.02       00:00:03.45
Command processing         106      00:00:00.35       00:00:02.75
Pass 1                     264      00:00:05.27       00:00:25.31
Symbol table sort            0      00:00:00.78       00:00:03.17
Pass 2                     138      00:00:01.41       00:00:05.55
Symbol table output         64      00:00:00.29       00:00:00.74
Psect synopsis output        2      00:00:00.02       00:00:00.02
Cross-reference output       0      00:00:00.00       00:00:00.00
Assembler run totals       605      00:00:08.14       00:00:41.00
```

The working set limit was 1650 pages.
49485 bytes (97 pages) of virtual memory were used to buffer the intermediate code.
There were 40 pages of symbol table space allocated to hold 757 non-local and 43 local symbols.
700 source lines were read in Pass 1, producing 23 object records in Pass 2.
17 pages of virtual memory were used to define 16 macros.

```
                              +------------------------------+
                              ! Macro library statistics !
                              +------------------------------+

Macro library name                      Macros defined
------------------                      --------------
_$255$DUA28:[MACRO.OBJ]MACRO.MLB;1             15
-$255$DUA28:[SYSLIB]STARLET.MLB;2               4
TOTALS (all libraries)                         19
```

851 GETS were required to define 19 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:ACTCHR/OBJ=OBJ$:ACTCHR MSRC$:ACTCHR/UPDATE=(ENH$:ACTCHR)+LIB$:MACRO/LIB

DEFINE
MAR

LOGINCMD
LIS

ACTONE
LIS

MESSAGES
LIS

ACTCHR
LIS

VALIDATE
LIS

ACTIF
LIS

PROTCLI
LIS

CRFMACROS
MAR

MACRO

MACRO32
MAP